Data-Efficient Performance Learning for Configurable Systems

Jianmei Guo

QCon October 17th 2017


极客时间
重拾极客精神·提升技术认知
每天10分钟，邀请顶级技术专家，为你传道授业解惑。
扫一扫，试读专栏


Geekbang
InfoQ
ArchSummit
全球架构师峰会 2017
12月8-9日 北京 · 国际会议中心

APSEC 2017

南京大学 × Geekbang> InfoQ


AiCon
全球人工智能技术大会 2018
助力人工智能落地
2018.1.13 - 1.14 北京国际会议中心
扫描关注大会官网

Team

Jianmei Guo
@Alibaba

Krzysztof Czarnecki
@Waterloo

Atrisha Sarkar
@Waterloo

Pavel Valov
@Waterloo

Sven Apel
@Passau

Norbert Siegmund
@Weimar

Andrzej Wąsowski
@Copenhagen

Configurable Systems are Ubiquitous

Configurable **software, hardware, human interaction**





Configurability → Flexibility

- Functional behavior

- Non-functional / quality properties
  - **performance**
  - cost
  - energy consumption
  - safety
  - security
  - etc.

Configure Software to Tailor Functional Behavior

a command-line tool to encode a video stream

```
x264 --quiet
     --no-progress
     --no-asm
     --rc-lookahead 60
     --ref 9
     -o trailer_480p24.x264
     trailer_2k_480p24.y4m
```

output stream

input stream

Configure Software to Tailor Functional Behavior

a command-line tool to encode a video stream

```
x264 --quiet
     --no-progress
     --no-asm
     --rc-lookahead 60
     --ref 9
     -o trailer_480p24.x264
     trailer_2k_480p24.y4m
```

feature1

feature2

feature3

feature4

feature5

configuration1
(variant1)

output stream

input stream

Configure Software to Meet a Certain Performance Goal

configuration1

```
x264 --quiet
     --no-progress
     --no-asm
     --rc-lookahead 60
     --ref 9
     -o trailer_480p24.x264
     trailer_2k_480p24.y4m
```

324 seconds

configuration2

```
x264
     --no-progress
     --no-asm
     --rc-lookahead 60
     --ref 9
     -o trailer_480p24.x264
     trailer_2k_480p24.y4m
```

551 seconds

Tuning only one option improves performance by 41% !

Goals

- Finding an **optimal configuration** to meet a specific performance goal

- Determining the **impact** of feature selections on performance

- Building the **performance model** behind a certain system

Measure the Performance of All Configurations?

- An exponential number of configurations
  - N Boolean features → about 2^N configurations
- The cost of measurement may be high
  - E.g., executing a complex benchmark

Feature-Wise Measurement?

configuration1

```
x264 --quiet
     --no-progress
     --no-asm
     --rc-lookahead 60
     --ref 9
     -o trailer_480p24.x264
     trailer_2k_480p24.y4m
```

324 seconds

configuration2

```
x264
     --no-progress
     --no-asm
     --rc-lookahead 60
     --ref 9
     -o trailer_480p24.x264
     trailer_2k_480p24.y4m
```

551 seconds

P(**quiet**)

= 551 – 324

= **227** seconds

Feature-Wise Measurement?

configuration1

```
x264 --quiet
     --no-progress
     --no-asm
     --rc-lookahead 60
     --ref 9
     -o trailer_480p24.x264
     trailer_2k_480p24.y4m
```

324 seconds

configuration2

```
x264
     --no-progress
     --no-asm
     --rc-lookahead 60
     --ref 9
     -o trailer_480p24.x264
     trailer_2k_480p24.y4m
```

551 seconds

P(**quiet**)

= 551 – 324

= **227** seconds

configuration3

```
x264 --quiet
     --no-progress

     --rc-lookahead 60
     --ref 9
     -o trailer_480p24.x264
     trailer_2k_480p24.y4m
```

487 seconds

configuration4

```
x264
     --no-progress

     --rc-lookahead 60
     --ref 9
     -o trailer_480p24.x264
     trailer_2k_480p24.y4m
```

661 seconds

P'(**quiet**)

= 661– 487

= **174** seconds

Feature-Wise Measurement?

configuration1

```
x264 --quiet
     --no-progress
     --no-asm
     --rc-lookahead 60
     --ref 9
     -o trailer_480p24.x264
     trailer_2k_480p24.y4m
```

324 seconds

configuration2

```
x264
     --no-progress
     --no-asm
     --rc-lookahead 60
     --ref 9
     -o trailer_480p24.x264
     trailer_2k_480p24.y4m
```

551 seconds

P(**quiet**)

= 551 – 324

= **227** seconds

Feature interactions

configuration3

```
x264 --quiet
     --no-progress

     --rc-lookahead 60
     --ref 9
     -o trailer_480p24.x264
     trailer_2k_480p24.y4m
```

487 seconds

configuration4

```
x264
     --no-progress

     --rc-lookahead 60
     --ref 9
     -o trailer_480p24.x264
     trailer_2k_480p24.y4m
```

661 seconds

P'(**quiet**)

= 661– 487

= **174** seconds

Key Challenges

- An exponential number of configurations
  - N Boolean features → about 2^N configurations
- The cost of measurement may be high
  - E.g., executing a complex benchmark
- Potential feature interactions
  - Hard to detect

Approaches

- Feature-interaction detection
  - [ICSE'12, SPLC'14, FSE'15, FSE'17]
- Non-linear regression
  - [ASE'13, SPLC'15, ASE'15a, ICPE'17]
- Fourier learning
  - [ASE'15b, SPLC'16]
  - ACM Distinguished Paper Award @ ASE'15
  - Best Paper Award @ SPLC'16

ICML 2016 Workshop on Data-Efficient Machine Learning

24 June 2016, Marriott Marquis (Astor Room), New York

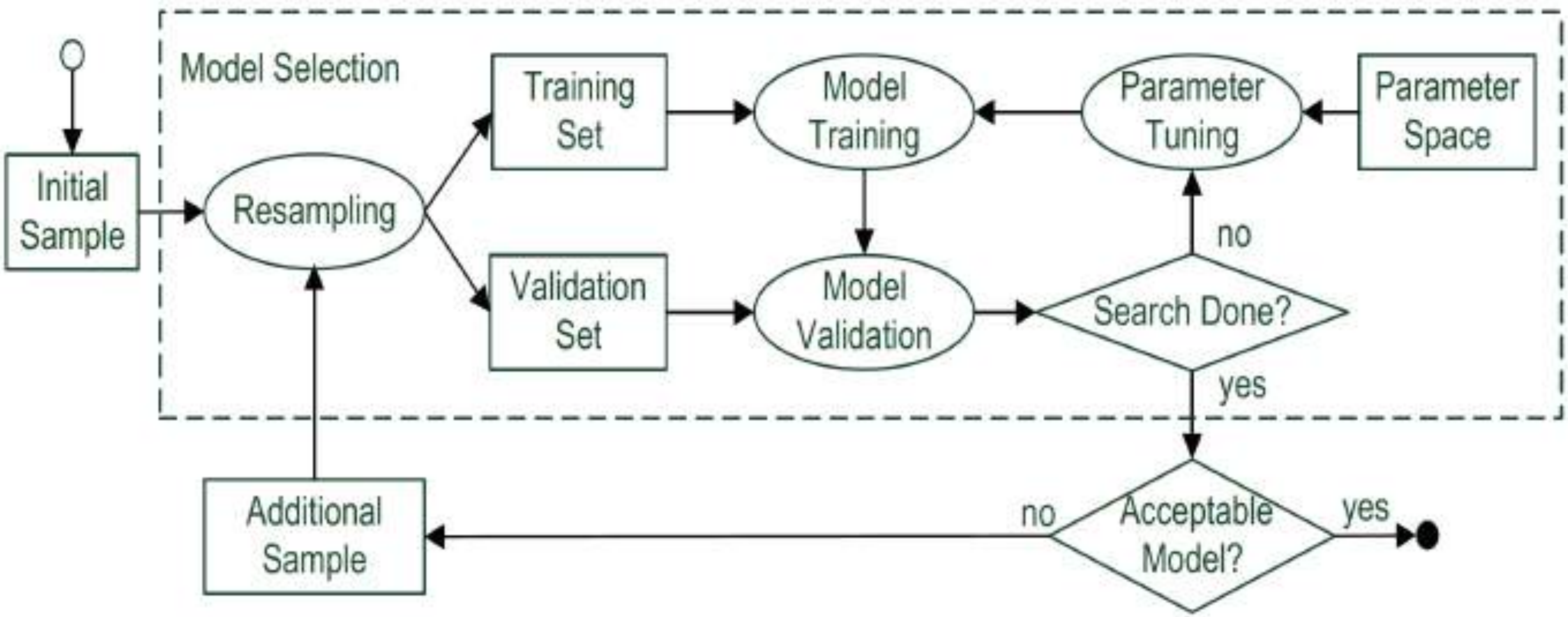
- Large-data problems (data is CHEAP)
  - object detection and recognition
  - machine translation
  - text-to-speech
  - recommender systems
  - information retrieval
- Small-data problems (data is EXPENSIVE)
  - personalized healthcare
  - robot reinforcement learning
  - sentiment analysis
  - community detection
  - **system quality prediction**

The ability to learn in complex domains without requiring large quantities of data!

Executive Summary

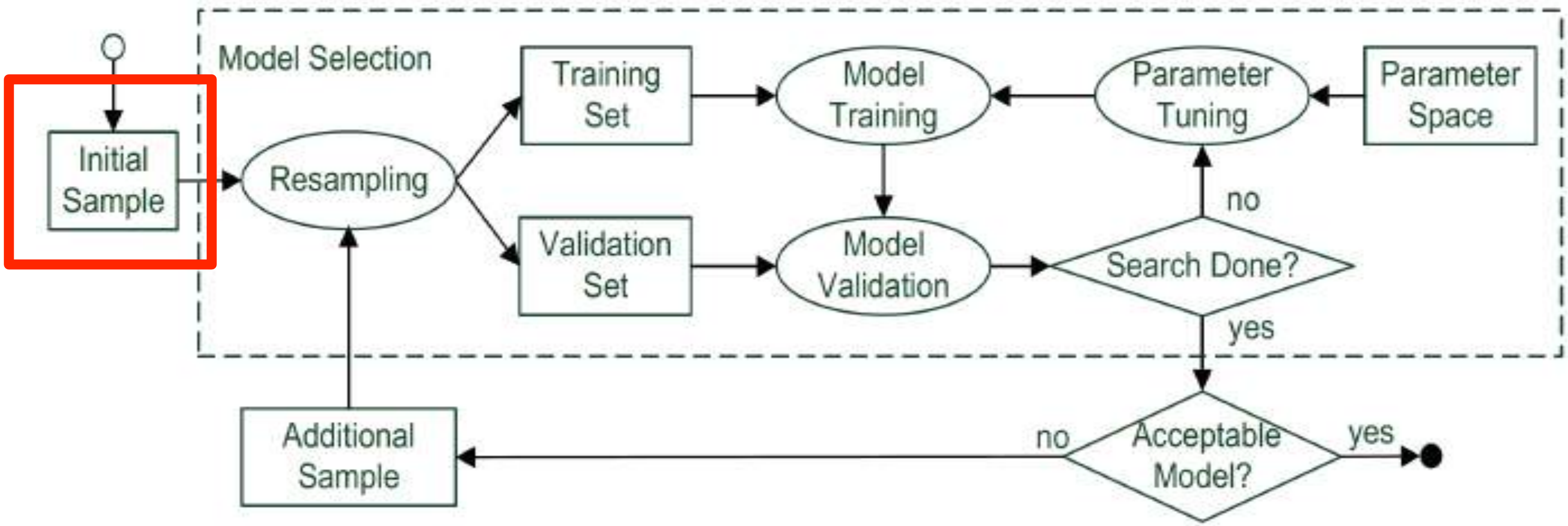
- Methodology
  - A well-established method CART
  - 3 resampling techniques
  - 3 parameter-tuning techniques
  - A sample quality metric
- Evaluation on 10 real-world systems
- Conclusion
  - DECART quickly (at most seconds) builds, validates, and determines an accurate (above 90%) performance prediction model based only on a given small sample of measured configurations, without additional effort to detect feature interactions
  - Ensures that the resulting model holds optimal parameter settings based on the currently available sample
  - Reaches a sweet spot between measurement effort and prediction accuracy
  - Works automatically and progressively with random samples of any sizes
  - Considers all features and identifies the performance-relevant ones
  - Easy to understand and easy to implement

DECART



Model Selection
Initial Sample
Resampling
Training Set
Validation Set
Model Training
Model Validation
Parameter Tuning
Parameter Space
Search Done?
no
yes
Acceptable Model?
no
yes
Additional Sample

Generate an Initial Sample

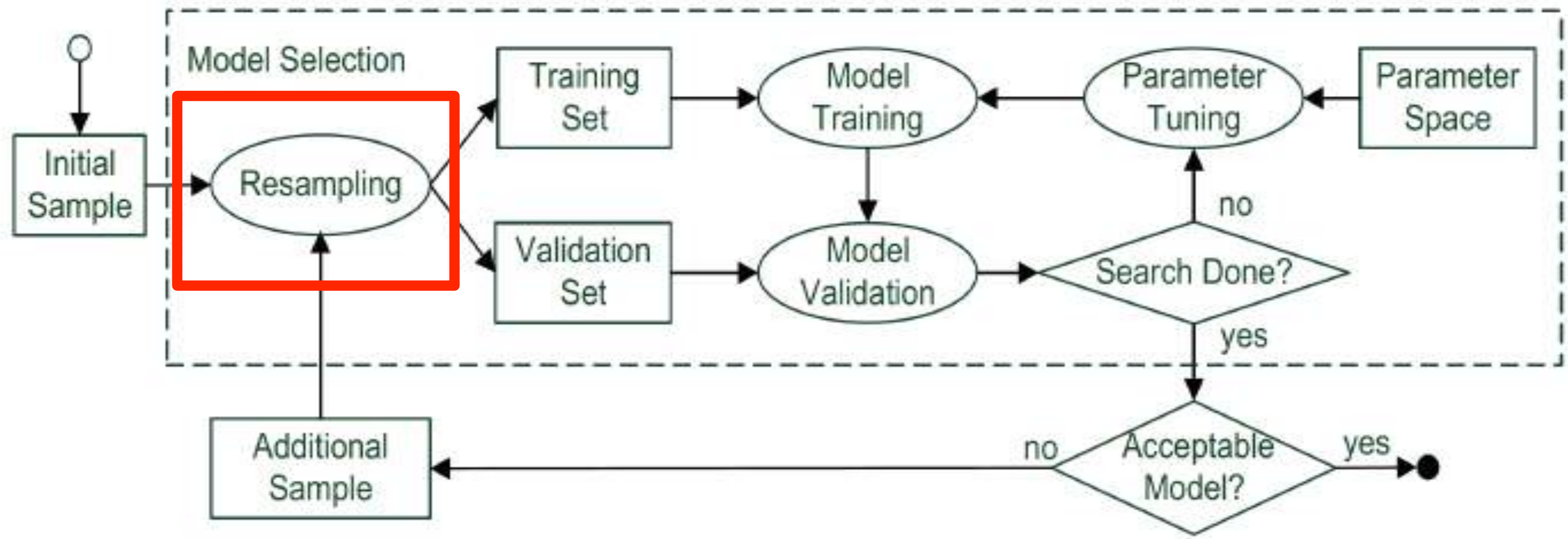


Three heuristics:

- feature-size: randomly selection, N
- feature-wise: simple coverage, N_W
- feature-frequency: combinatorial coverage, N_F

Roughly, $N < N_W < N_F$

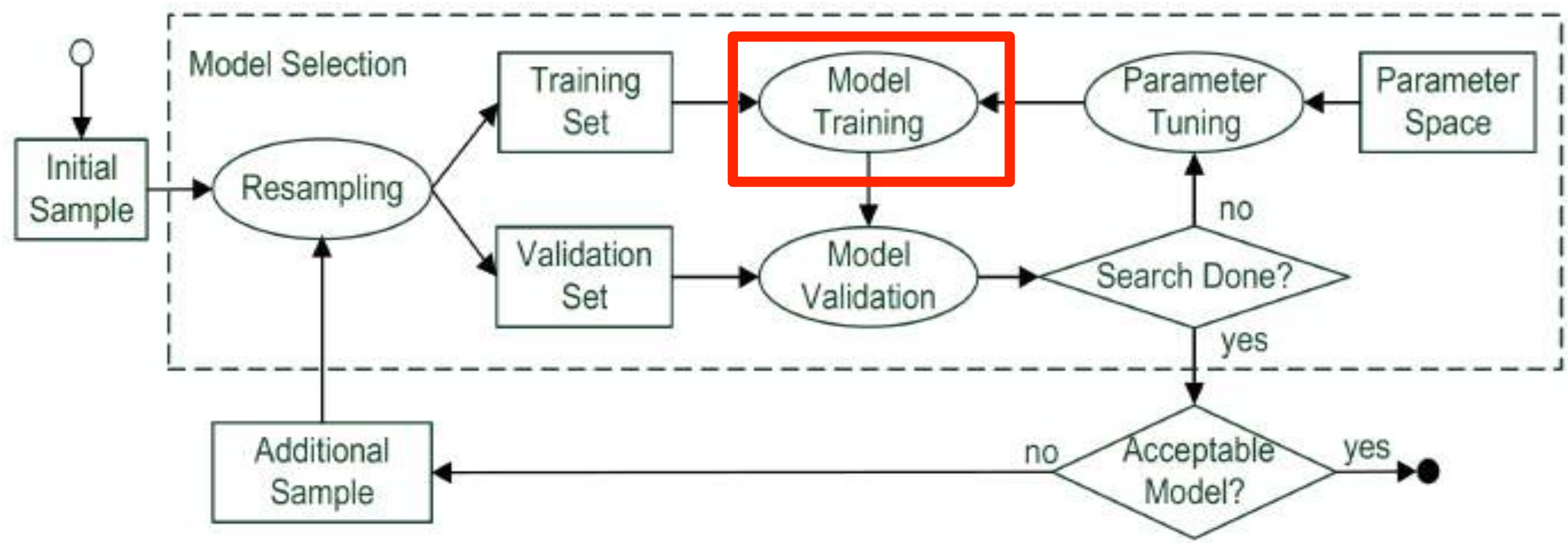
Resampling



Three well-established methods:

- hold-out
- k-fold cross-validation
- bootstrapping

Model Training



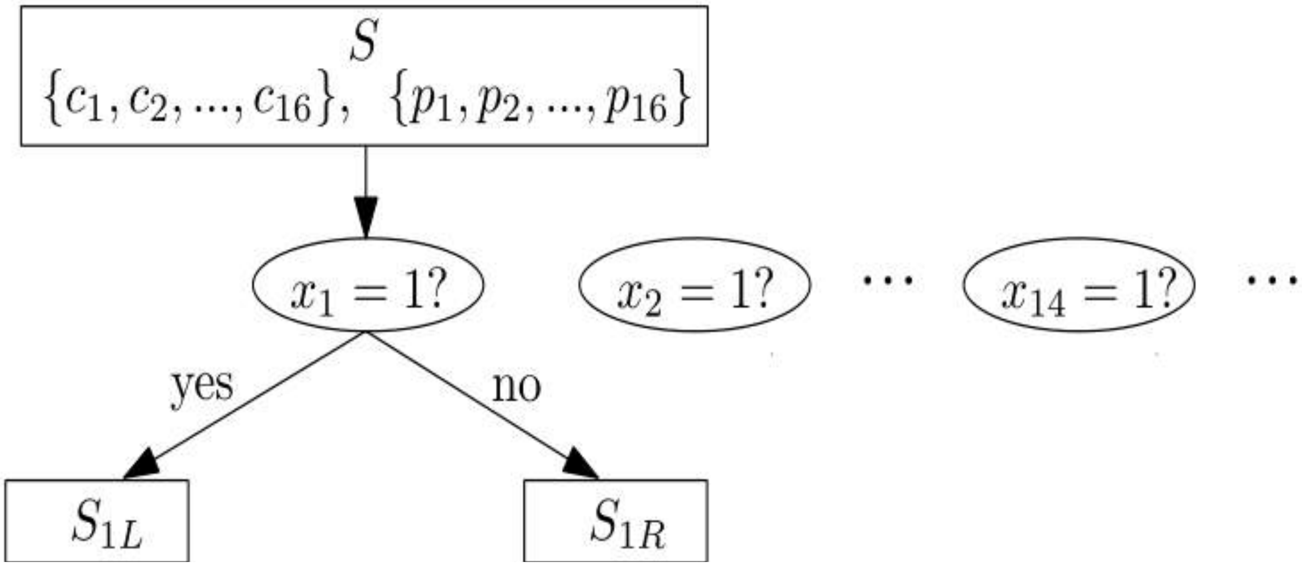
Classification And Regression Trees (CART)

- Non-linear learning
- Robust to noise data
- Usually very fast
- Easy to understand and easy to implement

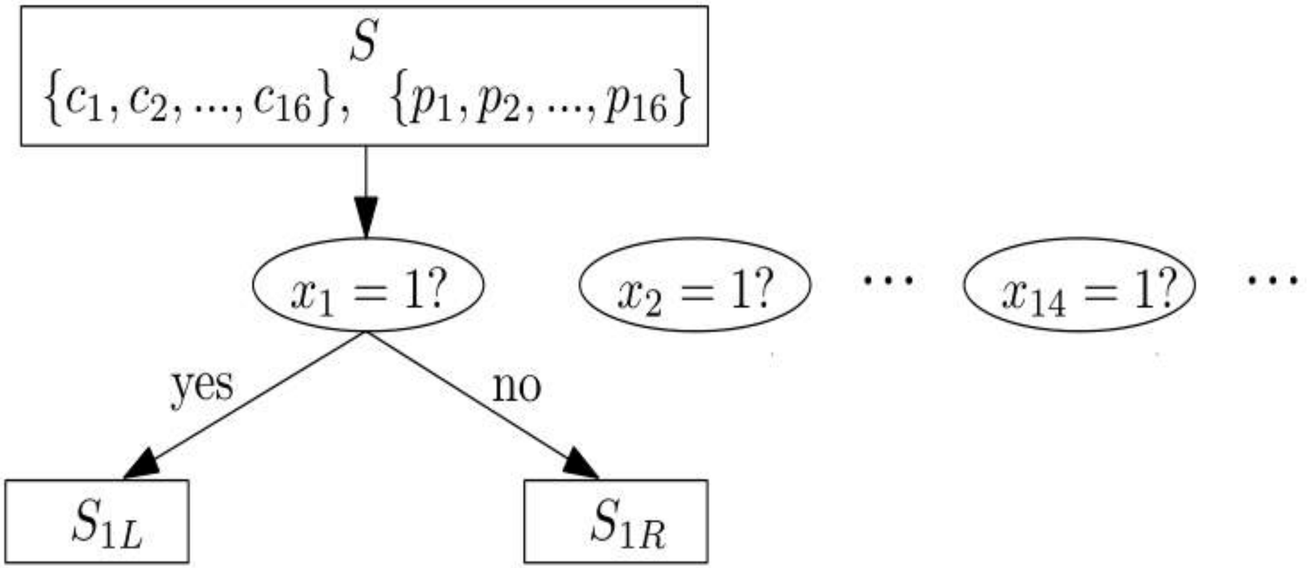
[Breiman et al., **Classification and Regression Trees. 1984**]

[Guo et al., **Variability-Aware Performance Prediction: A Statistical Learning Approach**. ASE'13]

CART

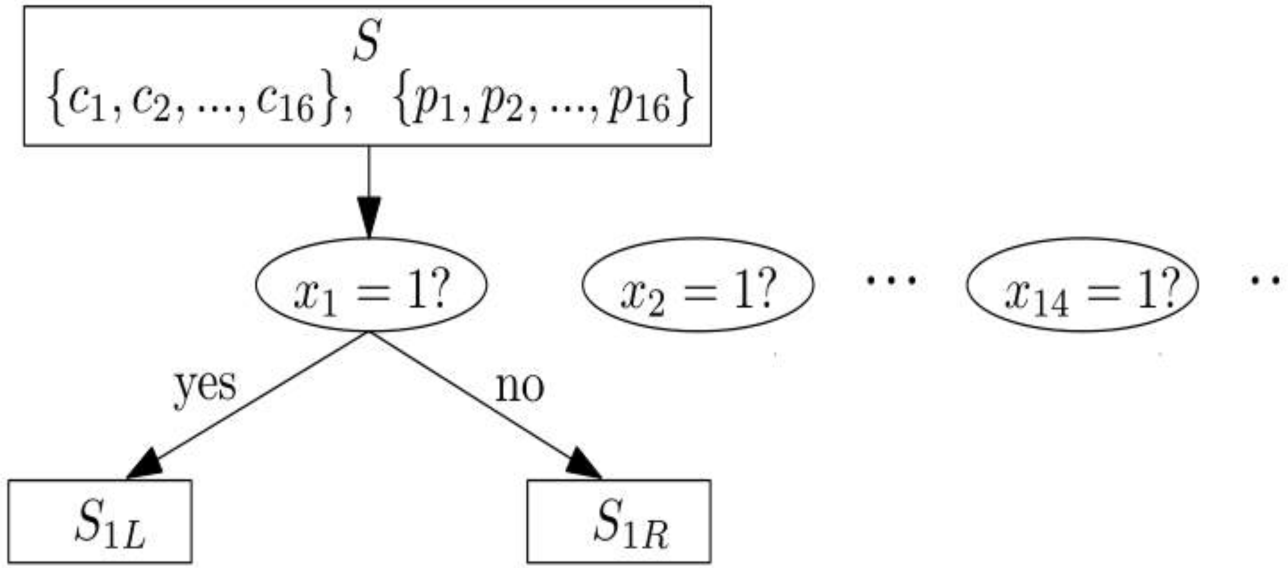


CART



Sample mean $\ell_{S_{iL}} = \frac{1}{|S_{iL}|}\sum_{p_j \in S_{iL}} p_j$ $\ell_{S_{iR}} = \frac{1}{|S_{iR}|}\sum_{p_j \in S_{iR}} p_j$

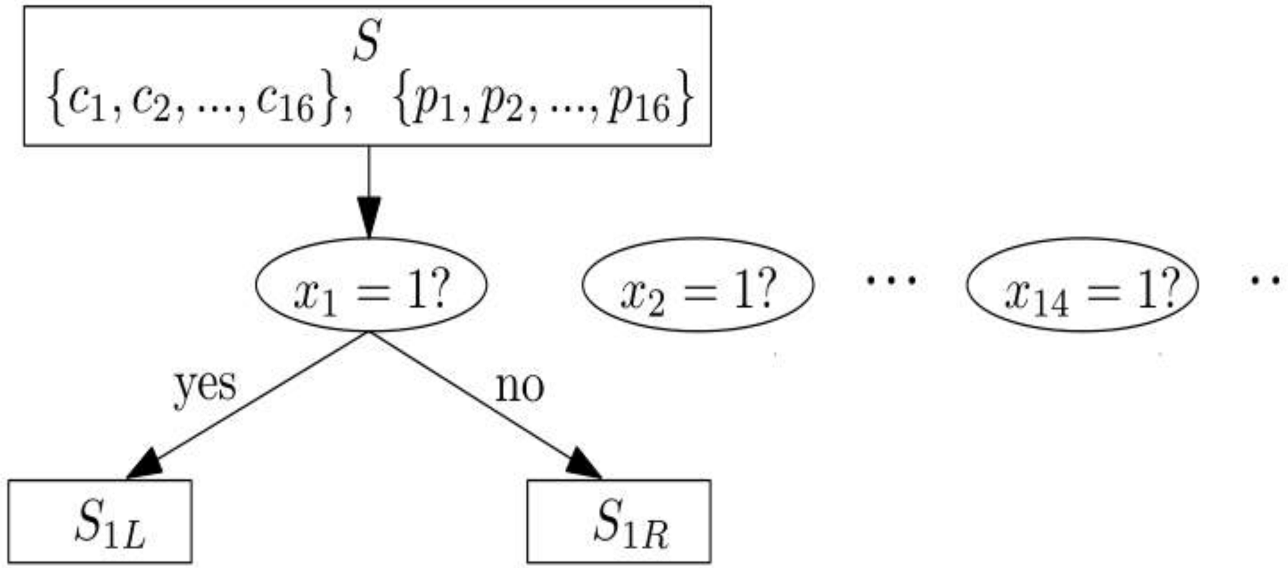
CART



Sample mean $\ell_{S_{iL}} = \frac{1}{|S_{iL}|} \sum_{p_j \in S_{iL}} p_j$ $\qquad$ $\ell_{S_{iR}} = \frac{1}{|S_{iR}|} \sum_{p_j \in S_{iR}} p_j$

Squared error loss $\sum_{p_j \in S_{iL}} (p_j - \ell_{S_{iL}})^2$ $\qquad$ $\sum_{p_j \in S_{iR}} (p_j - \ell_{S_{iR}})^2$

CART

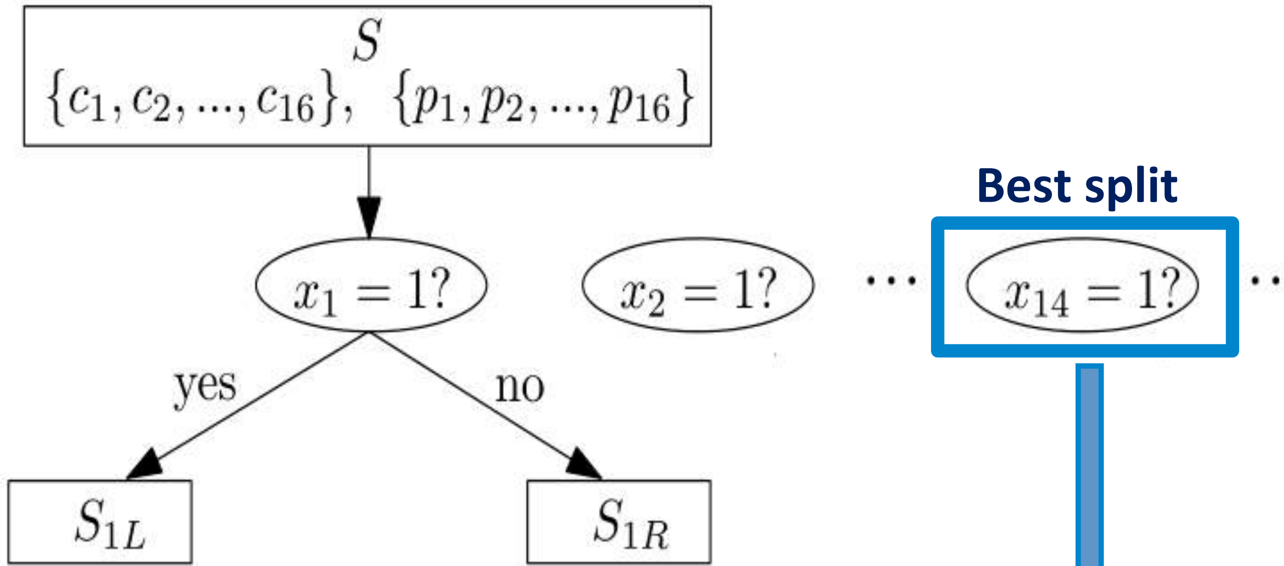


Sample mean $\ell_{S_{iL}} = \frac{1}{|S_{iL}|} \sum_{p_j \in S_{iL}} p_j$ $\quad \ell_{S_{iR}} = \frac{1}{|S_{iR}|} \sum_{p_j \in S_{iR}} p_j$

Squared error loss $\sum_{p_j \in S_{iL}} (p_j - \ell_{S_{iL}})^2$ $\quad \sum_{p_j \in S_{iR}} (p_j - \ell_{S_{iR}})^2$

Sum of squared error loss $\sum_{p_j \in S_{iL}} (p_j - \ell_{S_{iL}})^2 + \sum_{p_j \in S_{iR}} (p_j - \ell_{S_{iR}})^2$ **is minimal**

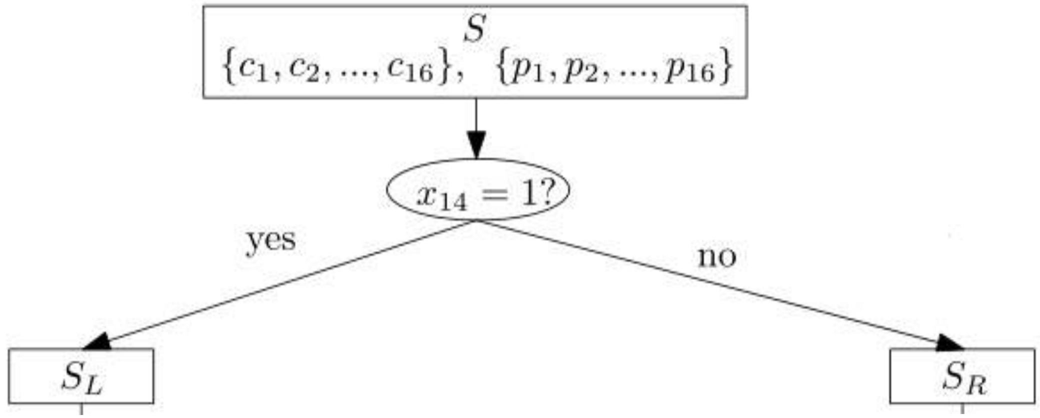
CART



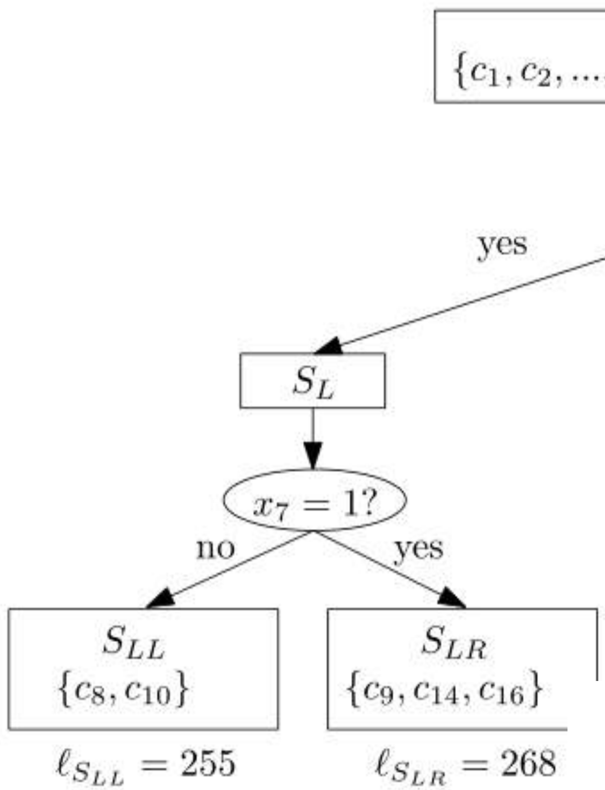
Sample mean $\ell_{S_{iL}} = \frac{1}{|S_{iL}|} \sum_{p_j \in S_{iL}} p_j$ $\quad$ $\ell_{S_{iR}} = \frac{1}{|S_{iR}|} \sum_{p_j \in S_{iR}} p_j$

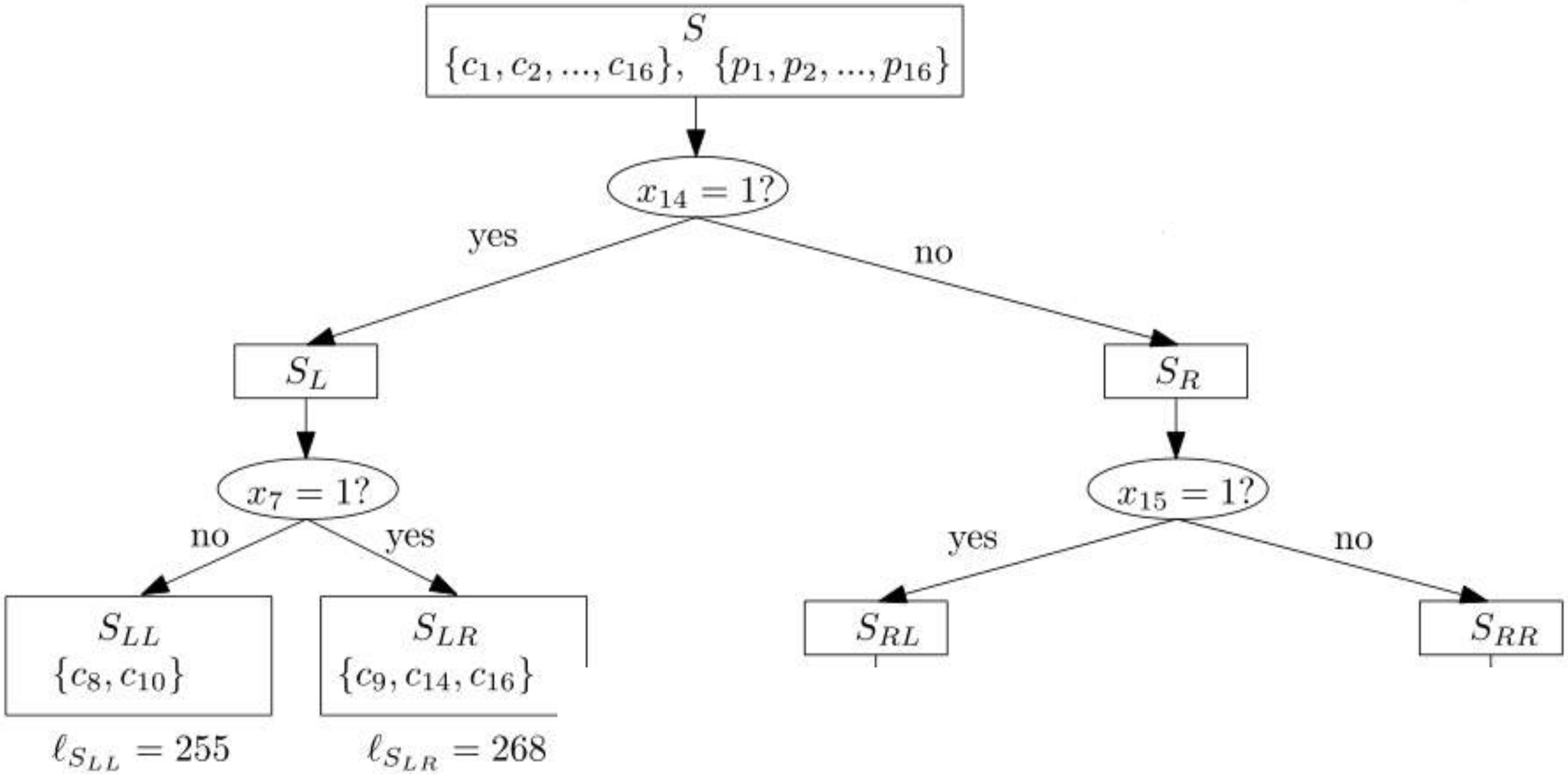
Squared error loss $\sum_{p_j \in S_{iL}} (p_j - \ell_{S_{iL}})^2$ $\quad$ $\sum_{p_j \in S_{iR}} (p_j - \ell_{S_{iR}})^2$

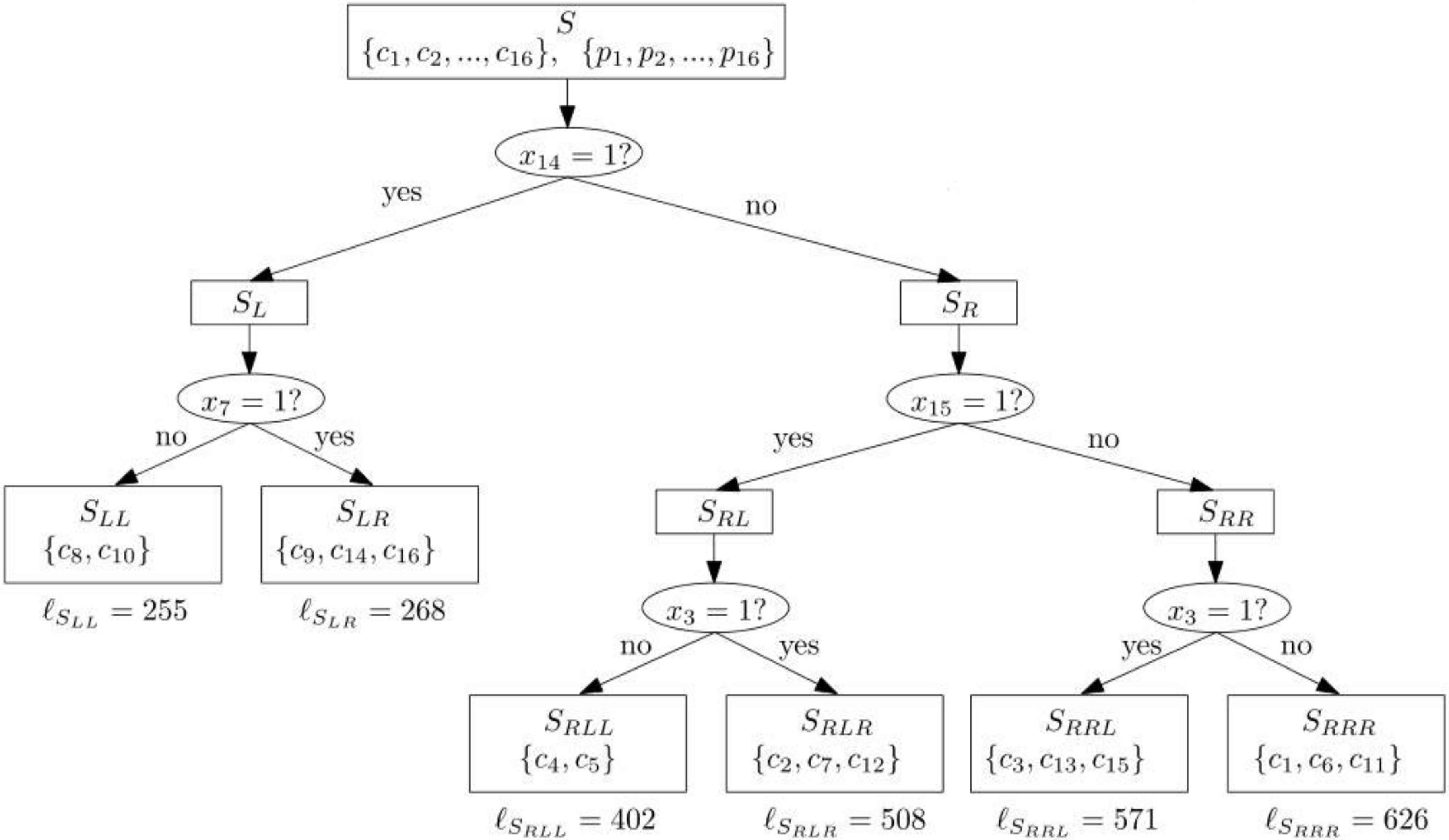
Sum of squared error loss $\sum_{p_j \in S_{iL}} (p_j - \ell_{S_{iL}})^2 + \sum_{p_j \in S_{iR}} (p_j - \ell_{S_{iR}})^2$ **is minimal**

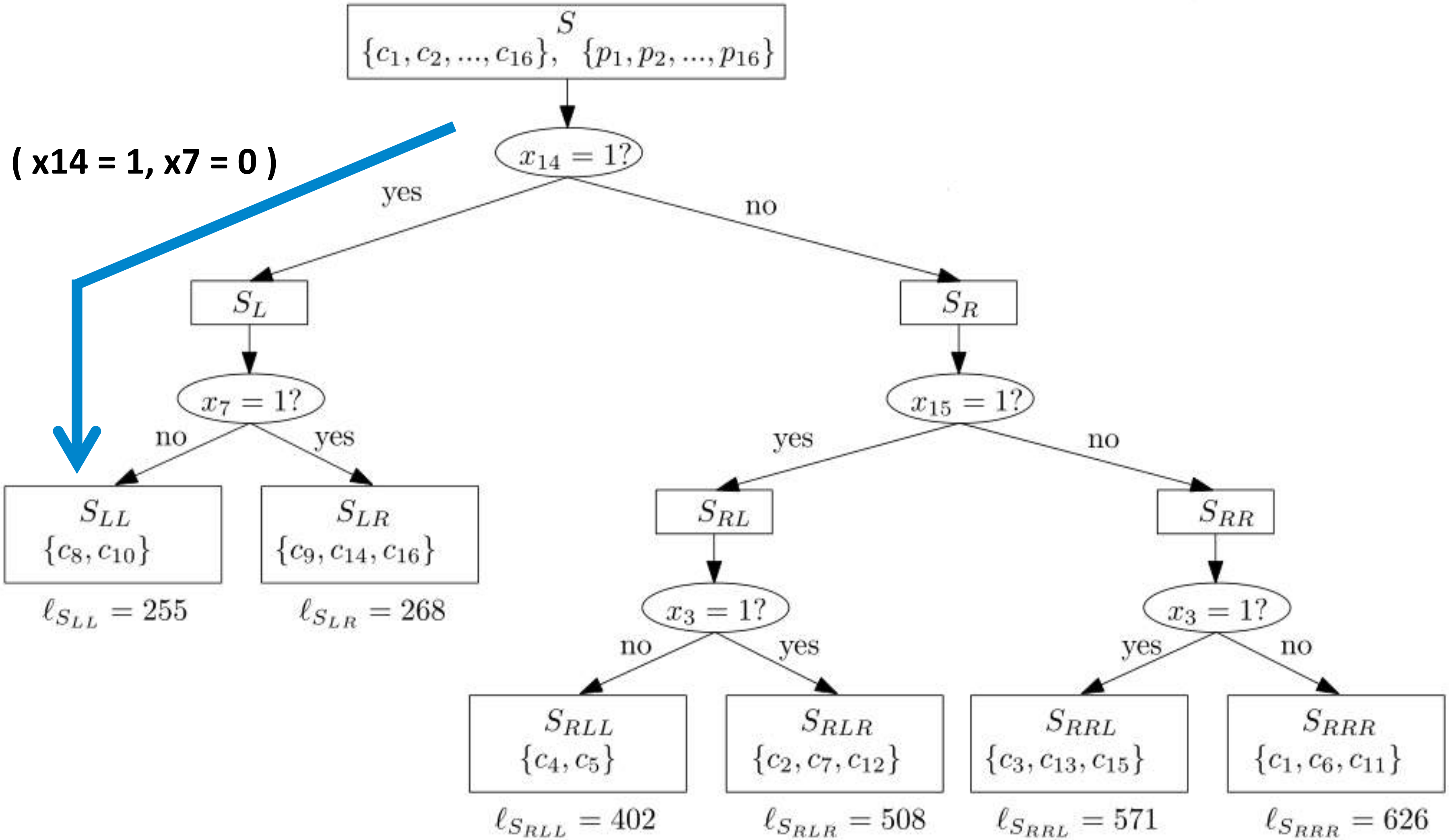


S
$\{c_1, c_2, ..., c_{16}\}, \quad \{p_1, p_2, ..., p_{16}\}$
$x_{14} = 1?$
yes
no
S_L
S_R

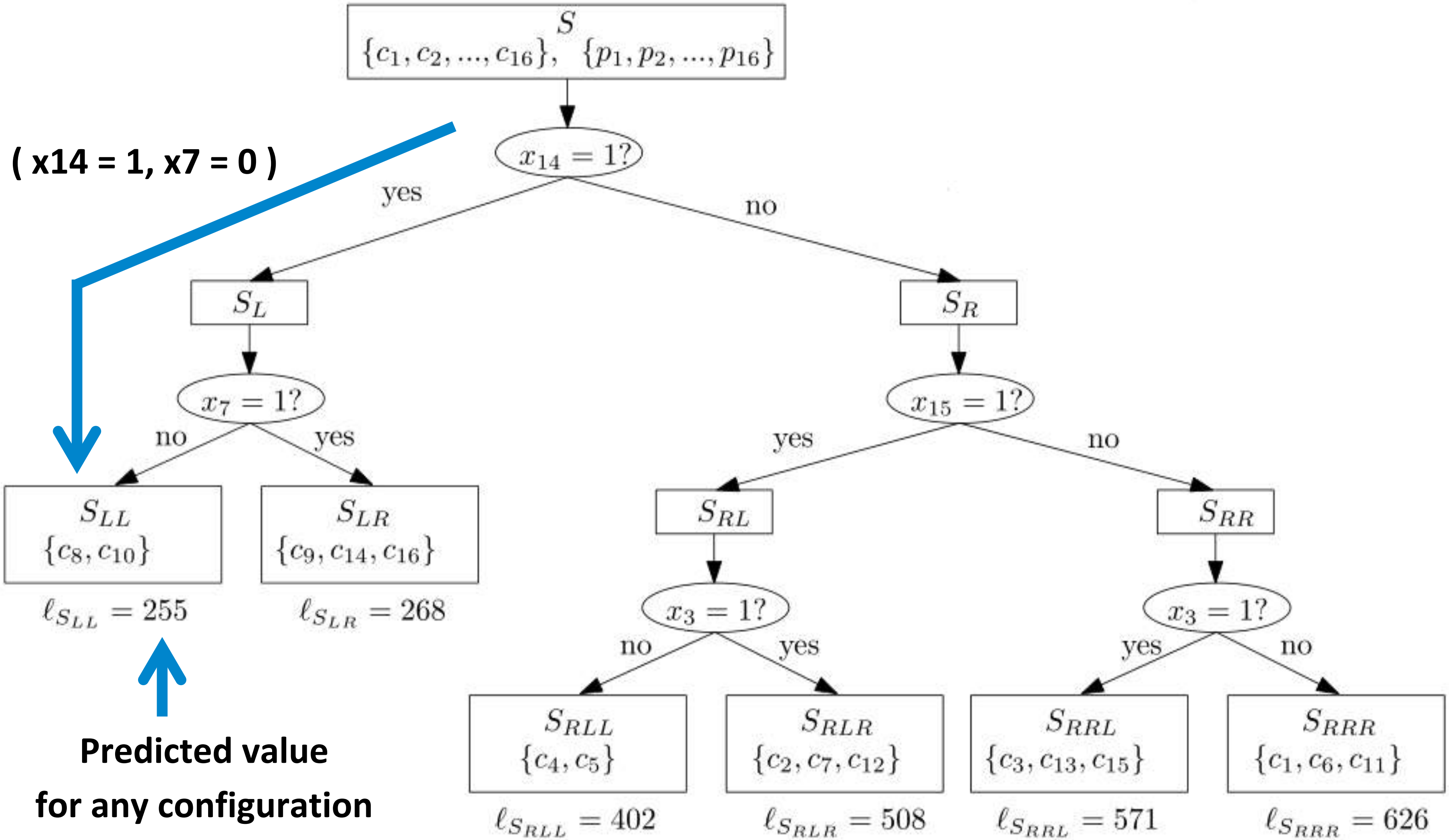

S
$\{c_1, c_2, ..., c_{16}\}, \quad \{p_1, p_2, ..., p_{16}\}$
$x_{14} = 1?$
yes
no
S_L
S_R
$x_7 = 1?$
no
yes
S_{LL}
$\{c_8, c_{10}\}$
S_{LR}
$\{c_9, c_{14}, c_{16}\}$
$\ell_{S_{LL}} = 255$
$\ell_{S_{LR}} = 268$


S
$\{c_1, c_2, ..., c_{16}\}, \quad \{p_1, p_2, ..., p_{16}\}$
$x_{14} = 1?$
yes
no
S_L
S_R
$x_7 = 1?$
$x_{15} = 1?$
no
yes
yes
no
S_{LL}
$\{c_8, c_{10}\}$
S_{LR}
$\{c_9, c_{14}, c_{16}\}$
S_{RL}
S_{RR}
$\ell_{S_{LL}} = 255$
$\ell_{S_{LR}} = 268$


S
$\{c_1, c_2, ..., c_{16}\}, \quad \{p_1, p_2, ..., p_{16}\}$
$x_{14} = 1?$
yes
no
S_L
S_R
$x_7 = 1?$
no
yes
S_{LL}
$\{c_8, c_{10}\}$
S_{LR}
$\{c_9, c_{14}, c_{16}\}$
$\ell_{S_{LL}} = 255$
$\ell_{S_{LR}} = 268$
$x_{15} = 1?$
yes
no
S_{RL}
S_{RR}
$x_3 = 1?$
no
yes
$x_3 = 1?$
yes
no
S_{RLL}
$\{c_4, c_5\}$
S_{RLR}
$\{c_2, c_7, c_{12}\}$
S_{RRL}
$\{c_3, c_{13}, c_{15}\}$
S_{RRR}
$\{c_1, c_6, c_{11}\}$
$\ell_{S_{RLL}} = 402$
$\ell_{S_{RLR}} = 508$
$\ell_{S_{RRL}} = 571$
$\ell_{S_{RRR}} = 626$

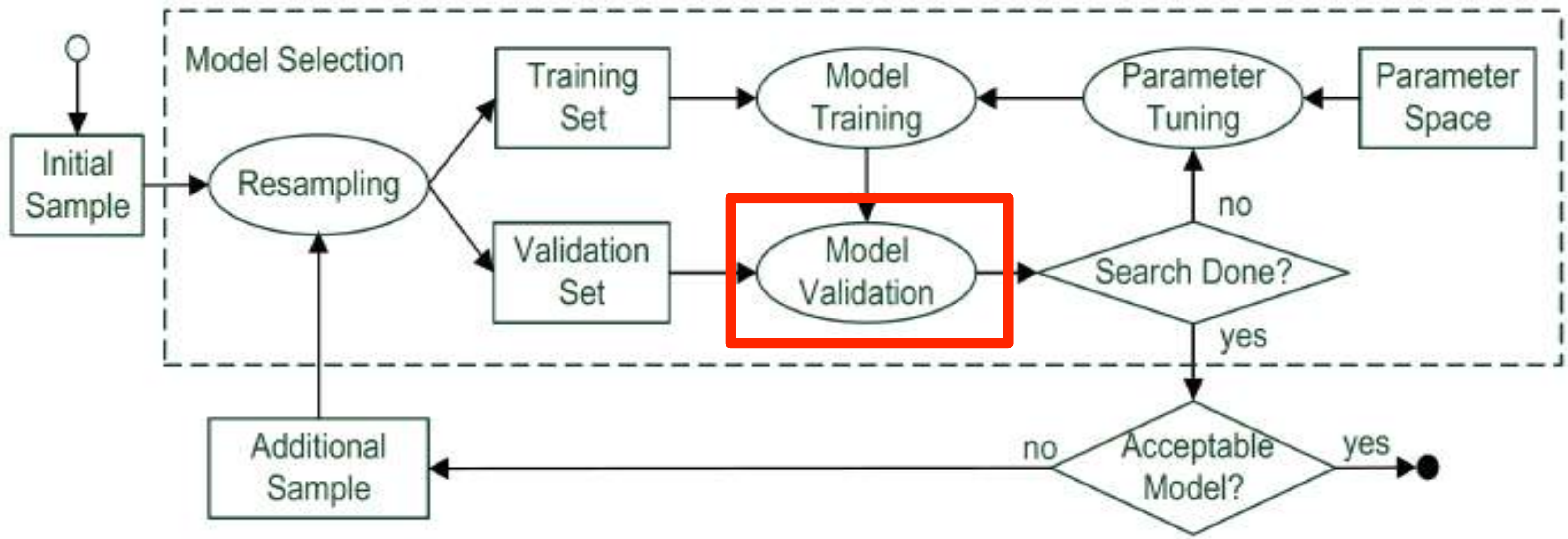

(x14 = 1, x7 = 0)
S
$\{c_1, c_2, ..., c_{16}\}, \quad \{p_1, p_2, ..., p_{16}\}$
$x_{14} = 1?$
yes
no
S_L
S_R
$x_7 = 1?$
no
yes
S_{LL}
$\{c_8, c_{10}\}$
$\ell_{S_{LL}} = 255$
S_{LR}
$\{c_9, c_{14}, c_{16}\}$
$\ell_{S_{LR}} = 268$
$x_{15} = 1?$
yes
no
S_{RL}
S_{RR}
$x_3 = 1?$
no
yes
$x_3 = 1?$
yes
no
S_{RLL}
$\{c_4, c_5\}$
$\ell_{S_{RLL}} = 402$
S_{RLR}
$\{c_2, c_7, c_{12}\}$
$\ell_{S_{RLR}} = 508$
S_{RRL}
$\{c_3, c_{13}, c_{15}\}$
$\ell_{S_{RRL}} = 571$
S_{RRR}
$\{c_1, c_6, c_{11}\}$
$\ell_{S_{RRR}} = 626$

(x14 = 1, x7 = 0)



Predicted value for any configuration selecting feature_14 and deselecting feature_7

Model Validation

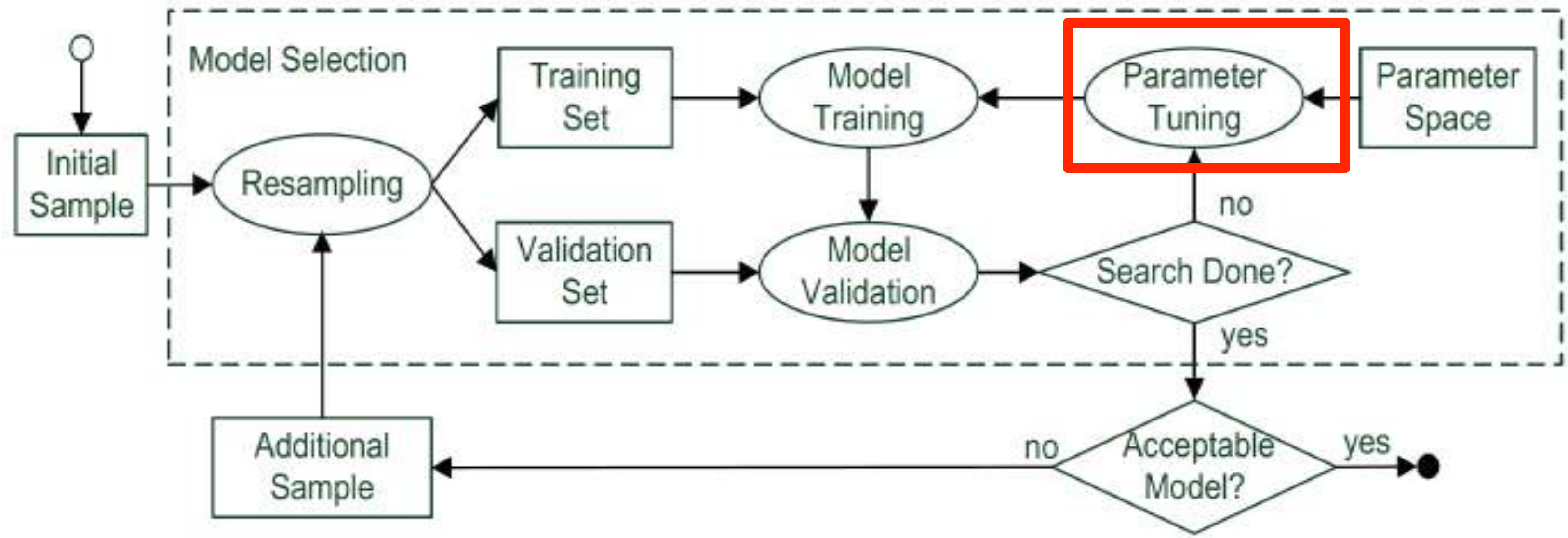


Mean Relative Error (MRE)

$$MRE = \frac{1}{|V|} \sum_{\mathbf{c} \in V} \frac{|actual_{\mathbf{c}} - predicted_{\mathbf{c}}|}{actual_{\mathbf{c}}}$$

Accuracy = 1 - MRE

Parameter Tuning



Determine the optimal parameter setting

- Define a parameter / hyper-parameter space
- Choose a parameter sweep method

Parameters vs. Hyper-Parameters

- A machine-learning **model** is the definition of a mathematical formula with a number of **parameters** that need to be **learned** from the data. By training a model with existing data, we are able to **fit** the model parameters.
- **Hyper-parameters** represent another kind of parameters that **cannot be directly learned** from the regular training process. These parameters express **"higher-level" properties** of the model, such as its complexity or how fast it should learn.

Parameter Space

Empirically determine a parameter space of CART by domain analysis

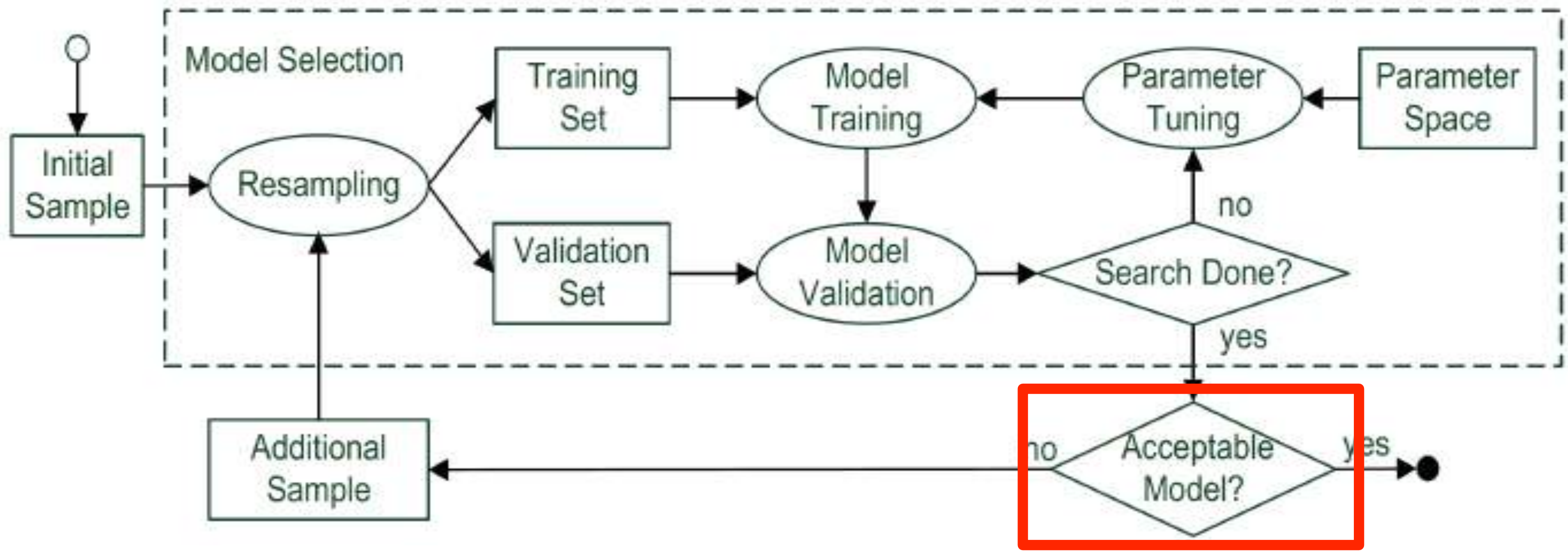
- Implementation in ***R***
  - *rpart(), rBayesianOptimization()*
- ***minsplit*** (integer): controls the minimum number of configurations that must exist in a tree node for further partitioning, [1, |S|]
- ***minbucket*** (integer): specifies the minimum number of configurations that must be present in any leaf node, *minbucket = minsplit / 3* [Williams, 2011]
- ***complexity*** *(real-value)*: controls the process of pruning a decision tree, and it is used to control the size of the tree and to select an optimal tree size, $[10^{-6}, 0.01]$

Parameter Sweep

Trade-off between exploration coverage and efficiency

- ***random search***: randomly tests a certain set of parameters
- ***grid search***: exhaustively tests all parameters
- ***Bayesian optimization***: uses a Gaussian Process to model the surrogate function that (1) is used to approximate the true performance function, and (2) it typically optimizes the expected improvement, which is the expected probability that new trials will improve on the current best observation

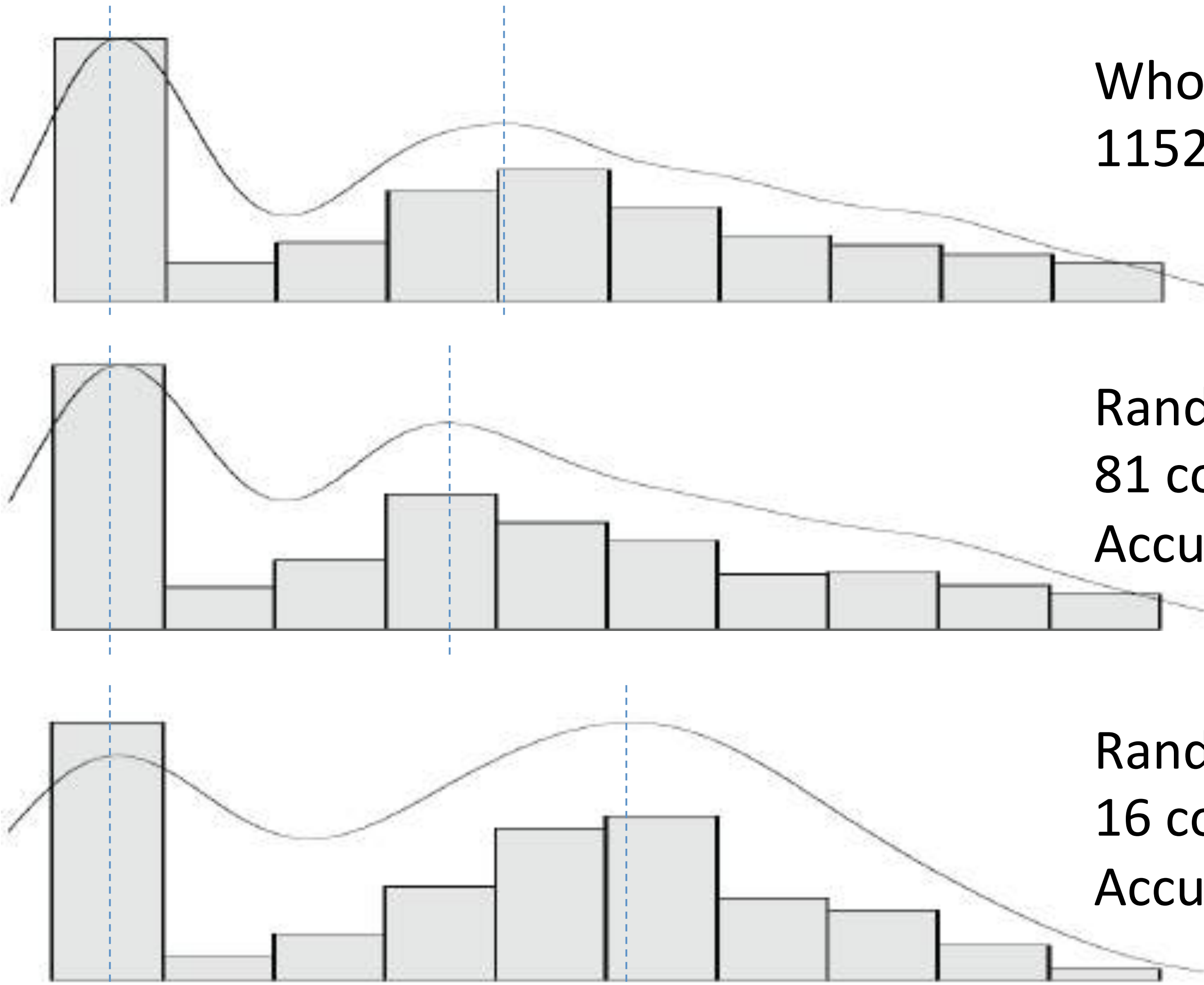
Stopping Criteria for Sampling



The key to the trade-off between measurement effort and prediction accuracy

- **Validation error**, calculated based only on the input sample *S*
- **Generalization error** on new data *WP* \ *S* (i.e., configurations not measured before)

Why CART works?



Whole population
1152 configurations

Random sample 1
81 configurations
Accuracy: 93.6%

Random sample 2
16 configurations
Accuracy: 84.9%

[Guo et al., **Variability-Aware Performance Prediction: A Statistical Learning Approach**. ASE’13]

Sample Quality Metric

Notoriously-known challenges in data mining

- Heterogeneous variables of different scales may give rise to unbalanced domination
  - Simple combination often makes numeric variables dominate
  - Normalized combination makes Boolean variables dominate

conf1: (f1=1, f2=1, f3=1, f4=1, f5=1), and performance = 1000s;
conf2: (f1=1, f2=1, f3=0, f4=0, f5=0), and performance = 1001s;
conf3: (f1=1, f2=0, f3=1, f4=1, f5=1), and performance = 10s;

Sample Quality Metric

- Measure a sample's distance or ***goodness of fit*** to the whole population by Pearson's Chi-squared test
- Key idea: sum up the differences between *observed* and *expected* outcome **frequencies** in terms of both feature selections and performance values

$$D_f(S,W) = \sum_{i=1}^{N} \frac{(O_{x_i}^S - E_{x_i}^S)^2}{E_{x_i}^S}$$

$$D_p(S,W) = \sum_{j=1}^{M} \frac{(O_{y_j}^S - E_{y_j}^S)^2}{E_{y_j}^S}$$

$$D(S,W) = \frac{D_f(S,W) + D_p(S,W)}{2}$$

Evaluation: Subjects

System	Domain	Language	LOC	N	$\|W\|$
AJSTATS	Code analyzer	C	14 782	19	30 256
APACHE	Web server	C	230 277	9	192
BDB-C	Database system	C	219 811	18	2 560
BDB-J	Database system	Java	42 596	26	180
CLASP	Answer set solver	C++	30 871	19	700
HIPA^{cc}	Video processing library	C++	25 605	52	13 485
LLVM	Compiler infrastructure	C++	47 549	11	1 024
LRZIP	Compression library	C++	9 132	19	432
SQLITE	Database system	C	312 625	39	4 653
x264	Video encoder	C	45 743	16	1 152

Comparing 3 Resampling Techniques

System	$\lvert S \rvert$	Bootstrapping				10-fold cross-validation				Hold-out			
		E_V (%)		Time (s)		E_V (%)		Time (s)		E_V (%)		Time (s)	
		Mean	±Margin	Mean	±Margin	Mean	±Margin	Mean	±Margin	Mean	±Margin	Mean	±Margin
AJSTATS	N	2.96	0.58	0.69	0.02	**0.60**	0.22	0.63	0.02	2.16	0.33	0.50	0.03
	$2N$	1.97	0.20	1.32	0.03	**1.49**	0.26	1.20	0.02	1.88	0.24	0.88	0.02
	$3N$	1.92	0.16	1.82	0.03	**1.35**	0.14	1.79	0.03	1.77	0.14	1.33	0.03
	$4N$	1.84	0.07	1.94	0.05	**1.47**	0.16	1.84	0.03	1.87	0.09	1.78	0.03
	$5N$	1.82	0.10	2.09	0.05	**1.49**	0.15	1.88	0.03	1.82	0.10	1.85	0.03
APACHE	N	17.87	3.3	0.22	0.00	–	–	–	–	**6.51**	27.73	0.15	0.00
	$2N$	17.79	3.01	0.45	0.01	5.8	2.30	0.42	0.01	**3.56**	14.95	0.34	0.01
	$3N$	9.36	1.35	0.67	0.00	5.72	1.18	0.65	0.01	**0.91**	10.19	0.47	0.01
	$4N$	8.44	0.61	0.88	0.00	6.40	1.59	0.85	0.01	**0.86**	9.48	0.68	0.02
	$5N$	6.83	0.57	1.14	0.01	5.16	0.61	1.05	0.01	**0.49**	8.83	0.82	0.02
BDB-C	N	89.6	32.25	0.59	0.00	**17.67**	7.84	0.60	0.02	76.38	29.19	0.38	0.01
	$2N$	36.38	8.91	1.21	0.03	**11.00**	5.99	1.08	0.02	27.82	11.46	0.77	0.01
	$3N$	19.04	4.06	1.76	0.04	**6.53**	2.59	1.65	0.03	15.88	6.33	1.15	0.01
	$4N$	11.63	3.12	1.83	0.04	**5.98**	2.28	1.74	0.04	6.32	1.12	1.63	0.02
	$5N$	6.36	1.44	1.87	0.04	**2.89**	0.92	1.83	0.04	5.71	1.14	1.71	0.03
BDB-J	N	4.70	2.33	0.87	0.01	**1.13**	0.29	0.81	0.02	8.05	4.02	0.61	0.01
	$2N$	1.76	0.20	1.80	0.02	**1.26**	0.24	1.67	0.03	1.71	0.17	1.23	0.01
	$3N$	1.79	0.13	1.88	0.03	**1.31**	0.19	1.93	0.02	1.57	0.16	1.8	0.02
	$4N$	1.58	0.12	1.92	0.04	**1.36**	0.22	1.93	0.02	1.66	0.14	1.96	0.05
	$5N$	1.56	0.10	1.95	0.02	**1.07**	0.13	1.99	0.02	1.46	0.13	1.89	0.02
CLASP	N	16.92	3.20	0.66	0.02	**7.03**	4.39	0.69	0.03	18.20	2.94	0.46	0.01
	$2N$	10.38	2.05	1.32	0.02	**5.31**	2.13	1.20	0.01	7.47	1.39	0.89	0.01
	$3N$	5.88	0.90	1.82	0.01	**1.93**	0.51	1.97	0.05	5.35	0.90	1.39	0.02
	$4N$	4.88	0.98	1.91	0.02	**2.05**	0.54	1.96	0.04	4.14	0.74	1.94	0.03
	$5N$	3.40	0.37	2.04	0.07	**2.07**	0.59	1.91	0.01	2.90	0.48	1.94	0.01
HIPA^{cc}	N	17.57	1.28	3.11	0.02	**12.52**	2.37	2.85	0.02	16.08	0.97	2.25	0.04
	$2N$	16.16	1.03	3.46	0.02	**12.94**	1.23	3.49	0.03	15.70	0.98	3.49	0.06
	$3N$	15.73	0.74	3.93	0.03	**11.63**	1.29	4.07	0.19	14.89	0.82	3.67	0.02

3 Parameter Tuning Methods

System	$\lvert S \rvert$	Bayesian optimization				Grid Search				Random Search			
		E_V (%)		Time (s)		E_V (%)		Time (s)		E_V (%)		Time (s)	
		Mean	±Margin	Mean	±Margin	Mean	±Margin	Mean	±Margin	Mean	±Margin	Mean	±Margin
AJSTATS	N	2.23	0.57	6.01	0.15	**0.60**	0.22	0.63	0.02	5.06	1.81	0.07	0.02
	$2N$	1.98	0.34	3.62	0.74	**1.49**	0.26	1.20	0.02	3.72	0.92	0.08	0.02
	$3N$	2.23	0.27	2.87	0.52	**1.35**	0.14	1.79	0.03	3.96	0.62	0.07	0.02
	$4N$	1.76	0.24	3.47	0.58	**1.47**	0.16	1.84	0.03	4.09	0.50	0.08	0.02
	$5N$	1.64	0.16	3.89	0.62	**1.49**	0.15	1.88	0.03	3.67	0.49	0.08	0.02
APACHE	N	–	–	–	–	–	–	–	–	–	–	–	–
	$2N$	9.43	3.44	6.1	0.49	**5.80**	2.30	0.42	0.01	29.62	4.94	0.02	0.00
	$3N$	6.65	1.41	7.62	0.61	**5.72**	1.18	0.65	0.01	20.12	4.21	0.01	0.00
	$4N$	7.67	0.97	2.80	0.48	**6.40**	1.59	0.85	0.01	29.48	2.85	0.01	0.00
	$5N$	6.76	1.06	5.57	0.19	**5.16**	0.61	1.05	0.01	32.37	2.56	0.01	0.00
BDB-C	N	34.05	10.17	7.09	0.21	**17.67**	7.84	0.60	0.02	483.09	210.08	0.02	0.00
	$2N$	29.39	11.67	2.78	0.46	**11.00**	5.99	1.08	0.02	624.47	148.14	0.02	0.00
	$3N$	28.82	5.23	4.92	0.94	**6.53**	2.59	1.65	0.03	35.37	7.75	0.02	0.00
	$4N$	21.57	3.50	3.94	0.81	**5.98**	2.28	1.74	0.04	73.08	35.56	0.02	0.00
	$5N$	20.30	2.57	4.56	0.80	**2.89**	0.92	1.83	0.04	35.81	3.12	0.02	0.00
BDB-J	N	1.65	0.58	7.60	0.92	**1.13**	0.29	0.81	0.02	26.32	15.74	0.01	0.00
	$2N$	2.40	0.36	2.76	1.04	**1.26**	0.24	1.67	0.03	41.06	14.02	0.02	0.00
	$3N$	1.95	0.35	5.48	0.17	**1.31**	0.19	1.93	0.02	30.17	7.26	0.02	0.00
	$4N$	2.01	0.26	5.24	2.23	**1.36**	0.22	1.93	0.02	11.97	8.51	0.02	0.00
	$5N$	1.96	0.21	5.71	1.61	**1.07**	0.13	1.99	0.02	3.00	0.17	0.02	0.00
CLASP	N	13.18	5.31	5.97	0.12	**7.03**	4.39	0.69	0.03	51.12	12.66	0.02	0.00
	$2N$	8.13	1.92	4.48	0.67	**5.31**	2.13	1.20	0.01	56.60	8.48	0.02	0.00
	$3N$	6.04	0.94	3.40	0.74	**1.93**	0.51	1.97	0.05	31.26	4.46	0.02	0.00
	$4N$	4.58	0.69	3.38	0.42	**2.05**	0.54	1.96	0.04	35.86	5.5	0.02	0.00
	$5N$	5.03	0.60	4.35	0.55	**2.07**	0.59	1.91	0.01	23.27	2.17	0.02	0.00
HIPA^{cc}	N	14.79	1.99	5.04	1.02	**12.52**	2.37	2.85	0.02	20.17	3.12	0.15	0.02
	$2N$	15.51	1.24	6.81	0.88	**12.94**	1.23	3.49	0.03	20.81	1.8	0.14	0.02
	$3N$	15.15	1.42	6.54	0.81	**11.63**	1.29	4.07	0.19	19.5	1.19	0.15	0.02

10-Fold Cross Validation + Grid Search

System	$\|S\|$	E_V (%)		E_G (%)		E_{CART} (%)		Time (s)		Q_S	
		Mean	±Margin	Mean	±Margin	Mean	±Margin	Mean	±Margin	Mean	±Margin
AJSTATS	N	0.60	0.22	3.41	0.49	2.94	0.50	0.63	0.02	1901.12	0.45
	$2N$	1.49	0.26	**2.77**	0.36	3.04	0.38	1.20	0.02	1892.72	0.45
	$3N$	1.35	0.14	**2.43**	0.31	2.72	0.41	1.79	0.03	1886.25	0.65
	$4N$	1.47	0.16	**2.13**	0.24	3.02	0.39	1.84	0.03	1877.07	0.67
	$5N$	1.49	0.15	**1.93**	0.06	2.73	0.34	1.88	0.03	1870.22	0.69
APACHE	N	–	–	–	–	–	–	–	–	–	–
	$2N$	5.80	2.30	14.68	2.32	11.32	1.25	0.42	0.01	20.66	0.38
	$3N$	5.72	1.18	**10.16**	1.04	10.23	1.04	0.65	0.01	18.29	0.44
	$4N$	6.40	1.59	**9.19**	1.06	9.51	0.97	0.85	0.01	16.13	0.43
	$5N$	5.16	0.61	8.99	0.94	8.64	0.74	1.05	0.01	14.57	0.49
BDB-C	N	17.67	7.84	147.65	56.25	123.13	37.55	0.60	0.02	1276.58	0.59
	$2N$	11.00	5.99	**56.84**	30.89	96.89	26.18	1.08	0.02	1268.23	0.59
	$3N$	6.53	2.59	**15.11**	3.54	77.31	19.12	1.65	0.03	1260.82	0.65
	$4N$	5.98	2.28	**8.06**	1.38	74.42	20.35	1.74	0.04	1252.70	0.65
	$5N$	2.89	0.92	**5.24**	0.91	59.92	12.40	1.83	0.04	1244.12	0.62
BDB-J	N	1.13	0.29	**3.22**	0.86	8.82	3.61	0.81	0.02	77.04	0.55
	$2N$	1.26	0.24	**2.07**	0.10	3.67	1.32	1.67	0.03	65.01	0.57
	$3N$	1.31	0.19	**1.85**	0.07	2.95	0.05	1.93	0.02	53.19	0.38
	$4N$	1.36	0.22	**1.68**	0.11	2.96	0.07	1.93	0.02	42.06	0.44
	$5N$	1.07	0.13	**1.67**	0.11	2.86	0.07	1.99	0.02	30.51	0.32
CLASP	N	7.03	4.39	15.33	6.91	22.82	2.36	0.69	0.03	279.00	0.80
	$2N$	5.31	2.13	**8.27**	1.28	17.61	1.00	1.20	0.01	270.32	0.83
	$3N$	1.93	0.51	**4.77**	0.89	18.62	1.17	1.97	0.05	261.11	0.58
	$4N$	2.05	0.54	**3.01**	0.40	18.65	1.13	1.96	0.04	253.37	0.81
	$5N$	2.07	0.59	**2.64**	0.29	17.53	0.93	1.91	0.01	245.34	0.69

Sweet-Spot between Measurement Effort and Prediction Accuracy?

System	$\|S\|$	$\frac{\|S\|}{\|W\|}$	E_V (%)		E_G (%)	
			Mean	±Margin	Mean	±Margin
AJSTATS	N	2×10^{-5}	0.60	0.22	3.41	0.49
APACHE	$2N$	9×10^{-2}	5.80	2.30	14.68	2.32
BDB-C	$3N$	2×10^{-2}	6.53	2.59	15.11	3.54
BDB-J	N	2×10^{-1}	1.13	0.29	3.22	0.86
CLASP	N	3×10^{-2}	7.03	4.39	15.33	6.91
HIPA^CC	$10N$	4×10^{-2}	9.46	0.55	10.32	0.42
LLVM	N	1×10^{-2}	1.81	0.76	5.97	0.24
LRZIP	$5N$	2×10^{-1}	6.67	1.61	11.72	2.05
SQLITE	N	1×10^{-5}	4.57	0.72	4.51	0.04
x264	N	1×10^{-2}	4.27	2.54	10.28	2.95

When the validation error is less than 10% ,
the generalization error also approximates to 10% !

Sweet-Spot between Measurement Effort and Prediction Accuracy?

System	$\|S\|$	$\frac{\|S\|}{\|W\|}$	E_V (%)		E_G (%)	
			Mean	±Margin	Mean	±Margin
AJSTATS	N	2×10^{-5}	0.60	0.22	3.41	0.49
APACHE	$2N$	9×10^{-2}	5.80	2.30	14.68	2.32
BDB-C	$3N$	2×10^{-2}	6.53	2.59	15.11	3.54
BDB-J	N	2×10^{-1}	1.13	0.29	3.22	0.86
CLASP	N	3×10^{-2}	7.03	4.39	15.33	6.91
HIPA[CC]	$10N$	4×10^{-2}	9.46	0.55	10.32	0.42
LLVM	N	1×10^{-2}	1.81	0.76	5.97	0.24
LRZIP	$5N$	2×10^{-1}	6.67	1.61	11.72	2.05
SQLITE	N	1×10^{-5}	4.57	0.72	4.51	0.04
x264	N	1×10^{-2}	4.27	2.54	10.28	2.95

When the validation error is less than 10% ,
the generalization error also approximates to 10% !

Conclusion

DECART: A **data-efficient** performance learning approach via **well-established** statistical learning techniques for configurable systems

- quickly (at most seconds) builds, validates, and determines an accurate (above 90%) performance prediction model based only on a given small sample of measured configurations, without additional effort to detect feature interactions
- Employs systematic resampling and parameter tuning to ensure that the resulting model holds optimal parameter settings based on the currently available sample
- Learns an accurate prediction model with as little measurement effort as possible for a given system, such that a sweet spot between measurement effort and prediction accuracy is reached
- Works automatically and progressively with random samples of any sizes
- Considers all features and identifies the performance-relevant ones
- Easy to understand and easy to implement

https://github.com/jmguo/DECART

Thank you for your attention!

QCon October 17th 2017